Feature Tracking in Images and Video

Anton Shokurov*, Andrey Khropov*, Denis Ivanov**
* Department of Mathematics and Mechanics, Moscow State University
** RL Labs Joint Stock Company, Moscow, Russia
{anton, akhropov}@fit.com.ru; denis@rl-labs.com

Abstract

Feature tracking is needed for many applications, for instance: 3d scene reconstruction, real and synthetic scene fusion, advanced video compression, super-resolution images creation and many others. In this paper we describe how we go about feature point tracking. We also describe feature point filtering, since some of the tracked points are actually outliers.

Keywords: *Feature Point Tracking, Feature Point Filtering, Fundamental Matrix, RANSAC.*

1. INTRODUCTION

In many applications a need of correct feature tracking arose. This problem has been intensively studied during the last decade. There are several main approaches and they can be generally divided into two main groups:

1) Direct methods try to compute a dense optical flow using intensity information from all pixels of the image.
2) Feature-based methods first extract features (i.e. points or area where meaningful information is concentrated) and then try to match only them.

For comprehensive survey of direct methods see [4] and for feature-based methods see [8].

Methods addressed in this paper are for static scenes. Feature tracking and filtering are done separately. First the feature tracker tracks some points, and then the filtering procedure removes outliers. We have chosen a feature-based strategy because we presumed a general assumption that the quality of input images might be relatively poor (as we have on ordinary camcorder) and might contain compression artifacts and therefore we should carefully choose information we may rely on.

2. FEATURE TRACKING

Our feature tracker is based on a cross-correlation approach to matching features in adjacent frames. We use this technique without multiscale strategy because we expect relatively small displacements between adjacent frames in our sequence (up to 10 pixels). One of the main disadvantages of this algorithm is that it is only pixel-precise. This may lead to appreciable errors if many frames have been processed. However, other approaches such as those described in [3],[5] demonstrated relatively worse performance on typical video sequences. Conceptually our tracking algorithm follows the method described in [1].

Our algorithm begins with a starting frame with some points selected on it and then during the processing of subsequent frames some of them disappear and therefore we also add points on a regular basis (every n-th frame) to maintain overall density. Each point has its initial frame where it has been added to the process. Then it is being tracked through the image sequence in the following manner:

- For each point we search for the best candidate in the next frame in the square neighborhood of the position of our point in the previous frame using a criteria of maximum cross-correlation of square neighborhoods in the current frame and in the initial frame for this point. We use an initial frame instead of the previous one to reduce discrepancy of found positions of feature points caused by numerical errors.

  $$\mathrm{CC(P_1,P_2)}=\frac{\sum\limits_{(k,l)\in NB} I^i_{i1+k,j1+l}\cdot I^c_{i2+k,j2+l}}{\sqrt{\sum\limits_{(k,l)\in NB}(I^i_{i1+k,j1+l})^2\cdot\sum\limits_{(k,l)\in NB}(I^c_{i2+k,j2+l})^2}}$$

  $\mathrm{P_1(i1,j1)}$ – point on an image with intensities I^i (initial)

  $\mathrm{P_2(i2,j2)}$ – point on an image with intensities I^c (current)

  $NB=\{(k,l)\in\mathbf{Z}^2 : k\in[-\mathrm{R},\mathrm{R}], l\in[-\mathrm{R},\mathrm{R}]\}$ - square neighborhood with a radius R

- If the best correlation score for a particular point is less than a predefined threshold we set the initial frame number for this point to an average of the old initial frame number and a current frame number and repeat the first step. We do this step iteratively until a satisfactory correlation score is reached or difference between the current frame number and the initial frame number becomes less than allowed (in such a situation this point is considered lost starting from the current frame and excluded from consecutive processing).
- We also repeat the correlation procedure as described in the previous item in the case when a displacement vector of the point is appreciably different from the weighted (according to distance to this point) average displacement of the surrounding points.

The result is presented in Figure 1.

3. FEATURE FILTERING

When feature tracking is done, not all of the geometrical constraints are used. For instance, the object being tracked is not deforming. For this reason given some points, we can constraint the movement of other points. For a two image filtering, an epipolar line is used: the point has to be at a certain distance form the line. For a three image filtering, a point can't be farther then a certain distance from a point, build using the epipolar lines from other two frames.

3.1 Building a Fundamental Matrix

We build a fundamental matrix using 7 points, as explained in [1], except that we handle differently cases when there are 3 real roots. Basically, we need to solve $(p^2{}_i,1)F(p^{1^i},1)=0$, given the corresponding points.

We can rewrite the equation under the following form: $[x^1x^2, y^1x^2, x^2, x^1y^2, y^1y^2, y^2, x^1, y^1, 1]f = 0, where$

$f = [F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}]^T$

Staking 7 equations, we get $Af = 0$. First we apply SVD to A, and get $USV^T f = 0 \Leftrightarrow Sg = 0, V^T f = g$. Last equation gives us a linear space of solutions: $f = \lambda_1 v^1 + \lambda_2 v^2$. We also note that the needed matrix is of rank-2. So we get constraint on this space of solutions: $\det f = 0$. From the last equation we get a cubic polynomial, which can be solved analytically. If we get one real solution, then we get the needed fundamental matrix. But if we get all 3 real solutions, then we test all three matrixes, after all our goal is to find the best fundamental matrix, not the points which where used to build the best fundamental matrix. [6] suggests us to sample more points in case of 3 real solutions. But by doing this we increase the sample size, and therefore we would need more samples in the RANSAC (RANdom SAmpling Consensus) algorithm [2].

3.2 Two Image Filtering

In this case we say that an outlier is a point, which is farther then a certain distance from an epipolar line, build using a corresponding point and the fundamental matrix. The best fundamental matrix is said to be the one, which has the most inliers. Following the RANSAC algorithm, we randomly sample our points. (Our sample consists of 7 points). Then we build the fundamental matrix (es) and count the number of inliers for each of them. We repeat this many times (like 1000), and the matrix left over is said to be the best.

3.3 Three Image Filtering

First lets say that a “true” point on third image, is a point which is build as an intersection of epipolar lines, build using the corresponding points on other images and the corresponding fundamental matrixes. Now we redefine an outlier as a point, which is farther then a certain distance from its “true” point on any of the images. The rest of the algorithm goes just like before, except that the sampled points are the same for all build fundamental matrixes: from image1 to image2, from image1 to image3, from image2 to image3.

3.4 Guided Sampling

The main idea behind this is that if we know that a certain point is an inlier, then we want to give it a better chance being randomly sampled. This is done, by assigning each point a weight. This weight is used when sampling. These weights can be dynamic, which means that their values change over time. We do it the following way. At all times we now the best (till this point) fundamental matrix, and the number of inliers for it. If a new fundamental matrix has more inliers then before, then the weight of all the inliers are increased, else nothing is done..

3.5 Filtering an Image Sequence

To filter out outliers from an image sequence, we process all consecutive 3 frames, starting from the first frame. After this process for each point we know on what frames it is an inlier and on what frames it is an outlier. We know where each point was defined. Now we define that a point is considered inlier on some set of adjacent frames, if on all of these frames the point is an inlier and the frame on which this point is defined is in this set and that we can't expand this set further to the left or right. On all other sets of frames the point is considered an outlier.

See filtering results in Figure 2.

4. CONCLUSION

The method presented in this paper was practically tested and verified its fitness for tracking feature points on video sequences captured by an ordinary camcorder in fully automatic mode.

We plan to improve this method to deal with more complicated cases where several independently moving objects are present (motion segmentation problem). Another objective is to track more complicated objects – lines and generic contours.

Filtering process can also be reformulated and generalized to the unified statistical framework for outlier rejection such as MLESAC [7].

The method presented in our work is not real-time but we designed it keeping a time factor in mind so it can be further improved to meet certain speed requirements.

5. ACKNOWLEDGMENTS

The research program presented in this paper is partially supported by the grant 03-07-90381 of Russian Foundation for Basic Research. Some results has been used in the “3D Head Calibration” project conducted by Computer Graphics Group at the Mathematics Department of MSU under a research agreement with Intel Nizhniy Novgorod Lab, which is a part of Intel Labs.

6. REFERENCES

[1] P. Beardsley, P. H. S. Torr, and A. Zisserman. *3d model aquisition from extended image sequences*. In B. Buxton and Cipolla R., editors, Proc. 4th European Conference on Computer Vision, LNCS 1065, Cambridge, pp. 683-695. Springer-Verlag, 1996.

[2] M. Fischler and R. Bolles. *Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography*. Commun. Assoc. Comp. Mach., vol. 24:381-95, 1981.

[3] B. K. P. Horn, B. G. Schunck. *Determining optical flow.* Artificial Intelligence, vol. 17, pp. 185-203, 1981.

[4] M. Irani, P. Anandan. *All about direct methods.* In W. Triggs, A. Zisserman, and R. Szeliski, editors, Vision Algorithms: Theory and practice. Springer-Verlag, 2000.

[5] B. D. Lucas, T. Kanade. *An iterative image registration technique with an application to stereo vision*. Proceedings of

the 7th International Joint Conference on Artificial Intelligence, Vancouver, pp. 674-679, 1981.

[6] M. Pollefeys, 3D Modeling from Images, tutorial notes, tutorial organized in conjunction with ECCV 2000, Dublin, Ireland, 26 June 2000. Newer online version: http://www.cs.unc.edu/~marc/tutorial.pdf

[7] P. H. S. Torr, A. Zisserman. *MLESAC: A new robust estimator with application to estimating image geometry*. Comp.Vision and Imag.Underst., vol. 78, pp. 138-156, 2000.

[8] P. H. S. Torr, A. Zisserman. *Feature-based methods for structure and motion estimation*. In W. Triggs, A. Zisserman, and R. Szeliski, editors, Vision Algorithms: Theory and practice. Springer-Verlag, 2000.

About the authors

Anton Shokurov is an undergraduate student at Moscow State University, Department of Mechanics and Mathematics. His contact email is anton@fit.com.ru.

Andrey Khropov is an undergraduate student at Moscow State University, Department of Mechanics and Mathematics. His contact email is akhropov@fit.com.ru.

Denis Ivanov works for RL Labs Joint Stock Company. He also leads some research projects at the Department of Mathematics and Mechanics of Moscow State University. His contact email is denis@rl-labs.com.

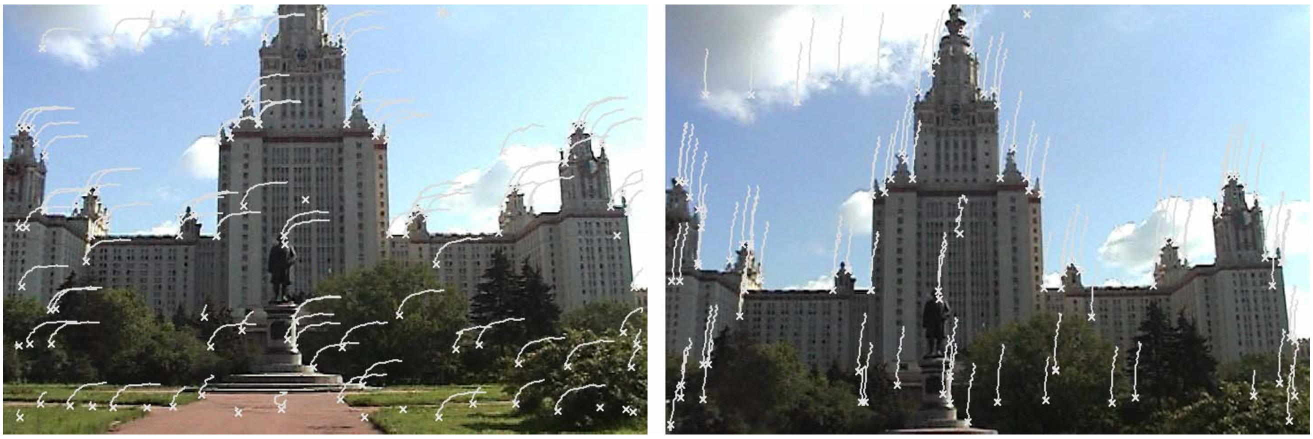

Figure 1. Two frames from sample video sequence with feature points shown with their trajectories.

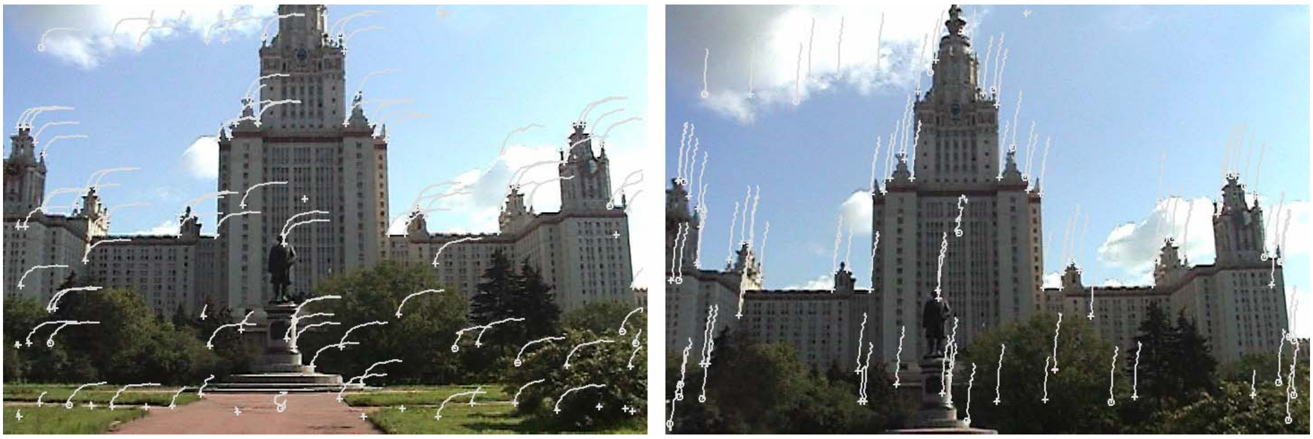

Figure 2. The same frames after filtering with inliers marked as crosses and outliers marked as circles.